Smart Air Quality Board for Pico

TECHNOLOGY DATA SHEET & SPECIFICATIONS

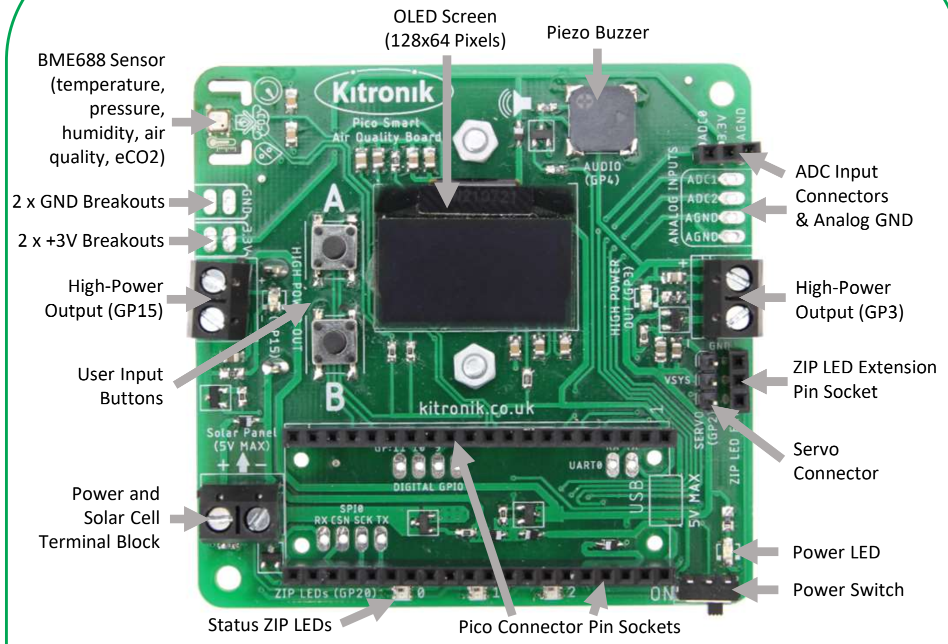
Introduction: **The Smart Air Quality Board provides a variety of sensor inputs and connection points for the Raspberry Pi Pico and provides the ability to log, store and display data effectively with the Pico filesystem and an OLED screen. There are also connection points for external devices, such as analog sensors, servos, motors, and heater pads.**

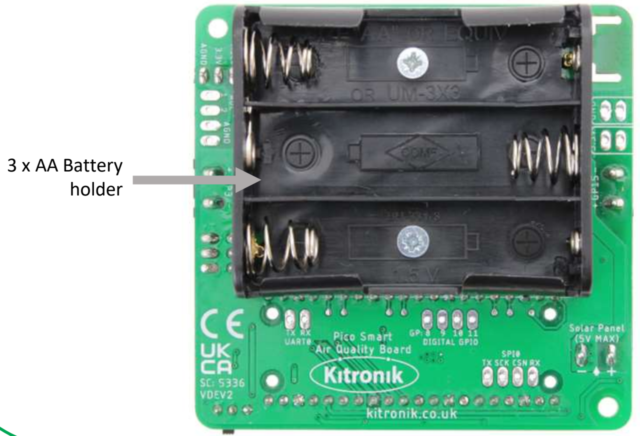
The board includes a dual row of pin sockets to connect your Raspberry Pi Pico. The Pico can then read inputs from a BME688 air quality and environmental sensor (temperature, pressure, humidity, air quality index and eCO2). The sensor, along with the RTC and flash memory on the Pico means the board is well setup for data logging. There is a black and white 128x64 OLED display screen and 3 status ZIP LEDs for visually displaying data, a piezo buzzer for audio and two buttons for user input. There are also external connections: 2 1A outputs, a servo output, an extension connection for more ZIP LEDs and analogue input connections linking to the Pico ADC (various other Pico pins are broken out to 0.1” pitch solder pads as further inputs and outputs, along with pads for 3V and GND).

Power is provided via the 3xAA battery holder or the ‘POWER’ terminal block. The voltage supply is controlled using a power switch, with a green LED to indicate when the board is turned on. The Pico also produces a **regulated 3V supply** which is used to power the BME688 sensor and OLED display screen.

The ‘POWER’ terminal block can also be used as the connection point for a solar cell.
NOTE: Please ensure the correct rechargeable batteries are fitted before charging, they should be **NiMh**.

Inserting a Pico: To use the Smart Air Quality Board the Pico should be inserted firmly into the connector (the image on the left shows the Pico in place ready to be pushed down).


OLED Screen (128x64 Pixels)
Piezo Buzzer
BME688 Sensor (temperature, pressure, humidity, air quality, eCO2)
2 x GND Breakouts
2 x +3V Breakouts
High-Power Output (GP15)
User Input Buttons
Power and Solar Cell Terminal Block
ADC Input Connectors & Analog GND
High-Power Output (GP3)
ZIP LED Extension Pin Socket
Servo Connector
Power LED
Power Switch
Status ZIP LEDs
Pico Connector Pin Sockets



Electrical Information

Operating Battery Voltage (Vcc)	+5V max
Power / Solar Cell Input	+5V DC max
Power Output Pins	2 x 3.3V, 2 x GND (250mA current draw max)
High-Power Output Terminal Blocks	GP3, GP15 (1A current draw max)
Servo Output Control	GP2 (Battery/Supply Voltage)
Piezo Buzzer Control	GP4
Analogue Inputs (Pico ADC)	ADC0, ADC1, ADC2 + 2 x Analogue GND
ZIP LED Control (including external extension)	GP20
Additional Pin Breakouts	Digital: GP8, 9, 10, 11; UART0: GP0, GP1; SPI0: GP16, 17, 18, 19
BME688 & OLED Display Control	GP6, GP7 (I2C1 Bus)
BME688 operating range	Pressure: 300hPa – 1100hPa Temperature: -40°C – 85°C Humidity: 0%RH – 100%RH Index for Air Quality (IAQ): 0 – 500 [0 = Excellent, 500 = Extremely Polluted] eCO2: 250 – 40000+ ppm [Estimated]

MicroPython Example Code

Kitronik have a MicroPython module to support the use of the Smart Air Quality Board. This can be found, along with example programs and a full README document, on the Kitronik GitHub:
https://github.com/KitronikLtd/Kitronik-Pico-Smart-Air-Quality-Board-MicroPython

This example code will help get you started:

The code on line 1 imports all the individual classes from the 'PicoAirQuality' module, and then lines 3 to 10 create instances of all these classes.

The BME688 gas sensor is setup on line 12, and line 13 carries out the required baseline readings and calculations ready for taking measurements.

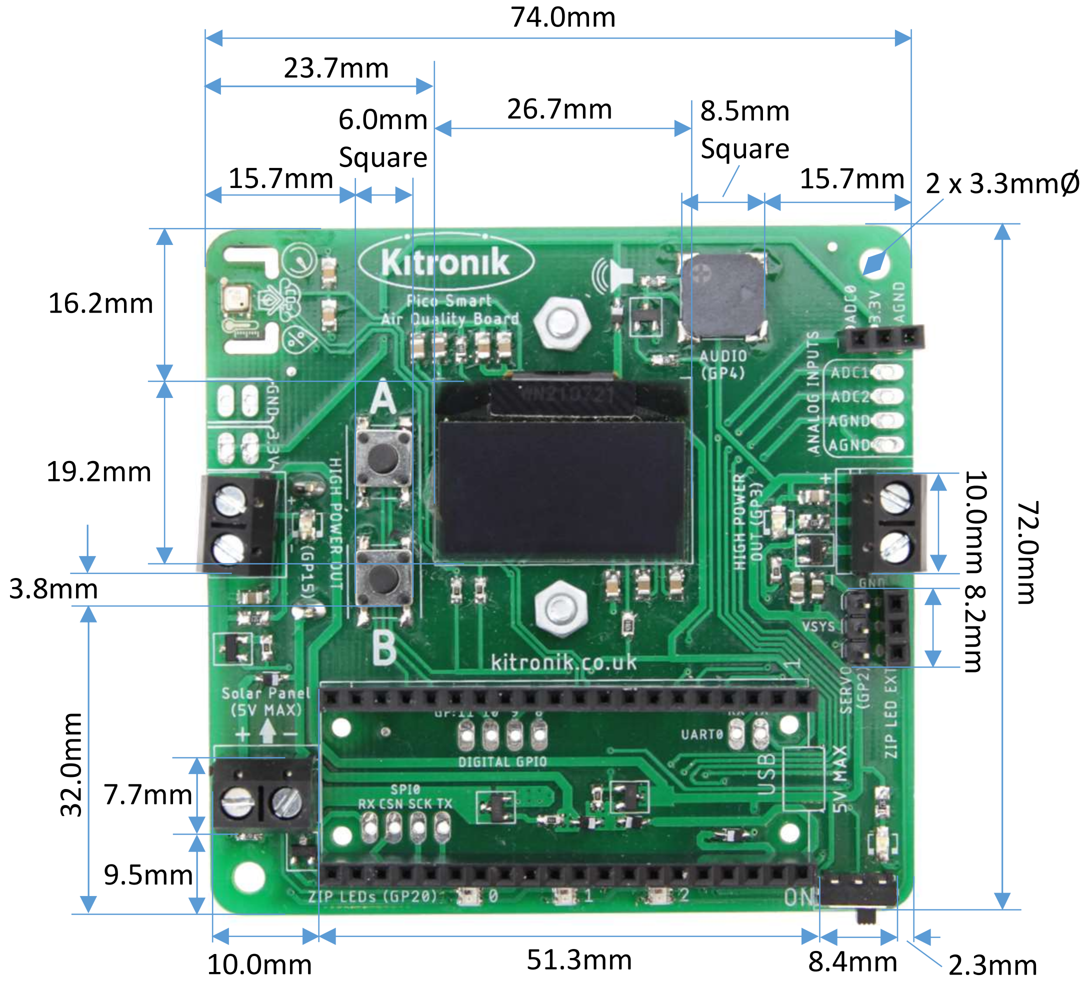
The 'while True' loop continually measures all the sensor readings from the BME688 and then prints these to the editor Shell (Thonny was used in the testing of this program – it is necessary to keep the Pico connected to the computer while the code is running).

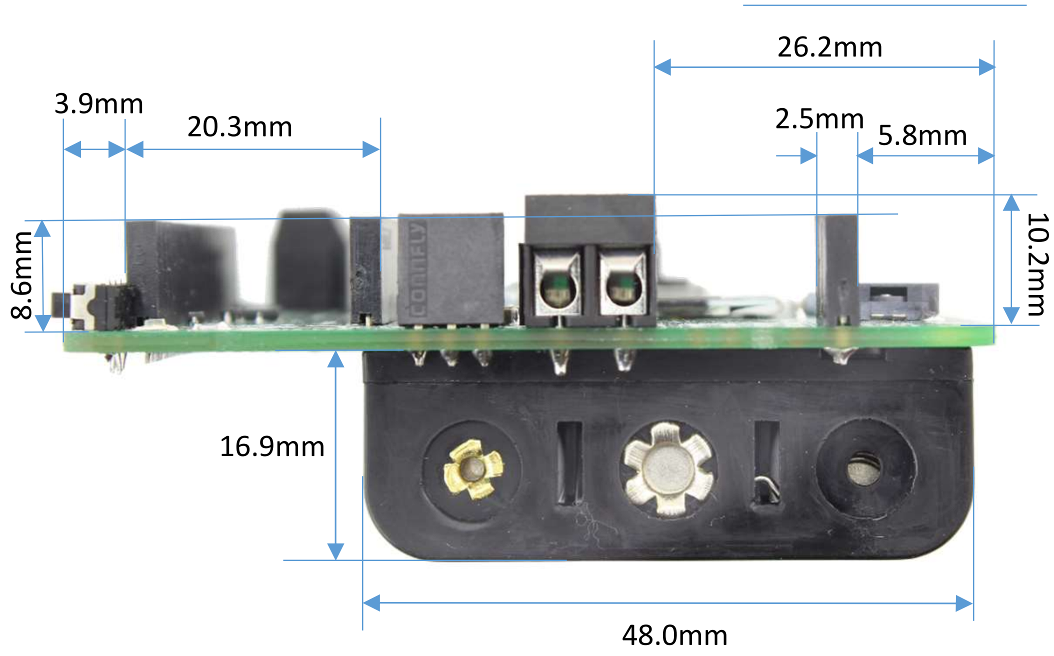
For more information on programming the Smart Air Quality Board visit:

kitronik.co.uk/5336

```
 1|from PicoAirQuality import KitronikBME688, KitronikOLED,
   KitronikRTC, KitronikZIPLEDs, KitronikBuzzer,
   KitronikDataLogger, KitronikOutputControl, KitronikButton
 2|
 3|bme688 = KitronikBME688() # Class for using the BME688 air
                              quality and environmental sensor
 4|oled = KitronikOLED() # Class for using the OLED display
                          screen
 5|rtc = KitronikRTC() # Class for using the built-in Pico
                         Real-Time Clock (RTC)
 6|zipleds = KitronikZIPLEDs(3) # Class for using the ZIP LEDs
                              (on-board and external connections)
 7|buzzer = KitronikBuzzer() # Class for using the piezo buzzer
 8|log = KitronikDataLogger("data_log.txt", "semicolon")
        # Class for using the Pico file system for data logging
 9|output = KitronikOutputControl() # Class for using the
                                     high-power and servo outputs
10|buttons = KitronikButton() # Class for using the input
                                buttons
11|
12|bme688.setupGasSensor()
13|bme688.calcBaselines()
14|
15|while True:
16|    bme688.measureData()
17|    print("Temperature: " + str(bme688.readTemperature())
                                                      + "C")
18|    print("Pressure: " + str(bme688.readPressure()) + " Pa")
19|    print("Humidity: " + str(bme688.readHumidity()) + " %")
20|    print("IAQ: " + str(bme688.getAirQualityScore()))
21|    print("eCO2: " + str(bme688.readeCO2()) + " ppm")
```

Dimensions





PCB Thickness: 1.6mm
(Dimensions +/- 0.2mm)